ELDERLY SUPPORT - ANDROID APPLICATION FOR FALL DETECTION AND TRACKING

By

TEJITHA RUDRARAJU

B.E, Anna University, India, 2011

A REPORT

Submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2014

Approved by:

Major Professor
Dr. Mitchell L. Neilsen

ABSTRACT

The purpose of the project is to develop an Android application that is capable of detecting possible falls for the elderly. With the advancement of wireless communications, the world has become smarter and there has been increase in use of smart phones. Android, being an open source platform has made it simple for every individual to develop their own applications, which in turn can be used on Android devices.

Falls among the elderly are a serious concern for both families as well as medical professionals, since falls are considered to be the eighth leading cause of death in the United States. Untreated fall injuries in adults 65 or older can result in serious risks and health complications, since 20% of falls require immediate medical attention and about one-tenth of the falls result in fractures. Thus, as mentioned, fall detection is a critical event requiring quick and accurate response, especially for elderly people living by themselves. This is the motivation behind developing an elderly support, which detects a fall and alerts the caretaker regarding the information.

The elderly support is about tracking the person and notifying the caretaker if there is an occurrence. One of the main features of this Android application is that it notifies the caretaker with alert messages which contain all the necessary information .The alert messages contain useful information about the people in danger, such as his/her geo location and also corresponding directions on a map. In occasions of false alerts, the supervised person is given the ability to estimate the value of importance of a possible alert and to stop it before proceeding with further steps. This project is geared towards supporting the elderly. The mobile application is capable of detecting possible falls and through a user-friendly interface that can be used to alert relatives, doctors, and other people who take care of the elderly.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank my family and friends for their unwavering support and rock-steady faith in me. I would not have made it without their constant encouragement and motivation.

I am deeply grateful to my major professor Dr. Mitchell L. Neilsen for his guidance, support, pointing me in the right direction, and for providing the Android devices for testing of my project. In addition, to his suggestions and guidance, I appreciate the trust he placed in me. I would like to thank Dr. Daniel Andresen and Dr. Doina Caragea for making their courses valuable and informative. It was a privilege to take courses under you, as they provided me with more practical guidance and experience. Thank you for graciously agreeing for serving on my committee.

Special thanks to Dr. Sandeep Pericherla as well, for the support he extended in difficult times and for being there for me when I needed him the most.

I would like to thank my friends - Anirudh, Swapna, Soumya, Swati, Prudvi, Aditya, Pavan, Kaushik, Priyanka, Bharath and Nikhita. Thanks for being with me through thick and thin.

Finally, thank you Asha Dharshini for being there for me and for your constant support and motivation.

Chapter 1 - INTRODUCTION

1.1 Project Description

Android is a Linux-based operating system for Android devices like Smartphones and Tablets. It is an open source platform available for users to develop Android applications using Android SDK (Software Development Kit). Elderly Support is an Android application that monitors the phone's built-in sensors, detects falls and generates pre-configured alerts on fall detection. The main aim of the project is to provide an easy and user-friendly way to alert the users during an occurrence. This makes it a fast and convenient first-aid tool.

The Google Maps API shows the location at which the fall occurred and also the nearest route to the specific location. The application also makes it easy for the user to setup and provide it to the elders before they head out or walk. The requirement analysis is discussed in Chapter 2, the motivation behind the project is discussed in Chapter 3, then followed by the system design in Chapter 4, Android framework in Chapter 5, implementation in Chapter 6, testing in Chapter 7, and finally the conclusion and future work of the project.

1.2 Motivation

The main motivation of this application is to learn the Android application development. Due to the rapid growth of mobile technology and the use of smartphones in day to day life, mobile technology was an interesting topic to initiate on. This led me on the path to develop an elderly-support Android application.

Falls increase risk for serious injuries, chronic pain, long-term disability, loss of independence and psychological and social limitations due to institutionalization. Nearly 50% of older adults are hospitalized for fall-related injuries. A fall can cause psychological damage even if the person did not suffer a physical injury. Those who fall often experience decrease activities of daily living and self-care due to fear of falling again. This behavior decreases their mobility, balance and fitness, leads to reduced social interactions, and increased depression. The mortality rate for falls increases progressively with age. In the age group of 65 and older, falls cause 57% of deaths due to injuries among females and 36% of deaths among males.

The main purpose of this project is to create an application for Android smart phones and tablets to aid the elderly community with a health problem, which can occur anywhere such as hospital/home/long-term, care institutions.

Chapter 2 - REQUIREMENTS ANALYSIS

2.1 Requirement Gathering

The development of the Android application involves requirements gathering, so that the prime features of the application can be accessible, extended and tested after the development.

The requirements of the application which were obtained after careful research are:

- GUI - use the important aspects of the Android application in a user friendly environment
- Security - provide security to information that is transferred over the network
- Server - to service the requests of devices
- Database - to store the information to be stored on the server and use while necessary

An Android application always involves analyzing the design so that we create an application that is more accessible and user friendly. In order to do so, the Android version, which is compatible with most of the Android devices, must be chosen and thus Android 2.3.3 Gingerbread version was chosen. Furthermore, the more detailed requirements that were required to develop the application are mentioned below.

2.2 Requirement Specification

The software and hardware requirements that are needed for developing the application are as follows

2.2.1Software Requirements

The software requirements for development and execution of application are mentioned,

For development of the application:

Operating System: Linux/Mac OS X 10.5.8 or higher /Windows XP or higher
Platform: Android SDK Framework 10 or higher
Java Database: SQLite Database
Tools: Eclipse SDK 3.5, ADT plug-in for eclipse
Technologies used: Java, SQLite, Android, Google Maps V2 API
Debugger: Android DDMS - Dalvik Debug Monitor Service
Android Emulator: SDK Version 3.0 or higher

For running the application on Android device:

Operating System: Android 3.0 or higher versions

2.2.2 Hardware Requirements

The hardware requirements for developing and execution of the application are mentioned below application.

For developing the application:

Processor: Intel Pentium IV or higher
RAM: 256 MB
Space on disk: minimum 250MB

Hardware Requirements for running the application:

Device: Smart Phone or Android Tablet with Android version 3.0 or higher
Minimum space to execute: 5.0MB

Chapter 3 - BACKGROUND AND RELATED WORK

3.1 Android OS

Android is an operating system, which is based on Linux kernel which can be used on mobile devices. It is open source and can be used to create new applications and also to modify existing applications. The developed applications can be distributed to other Android users over the Android market. The immense amount of documentation available online makes it uncomplicated to develop an application in the Android operating system.

3.2 Android Architecture

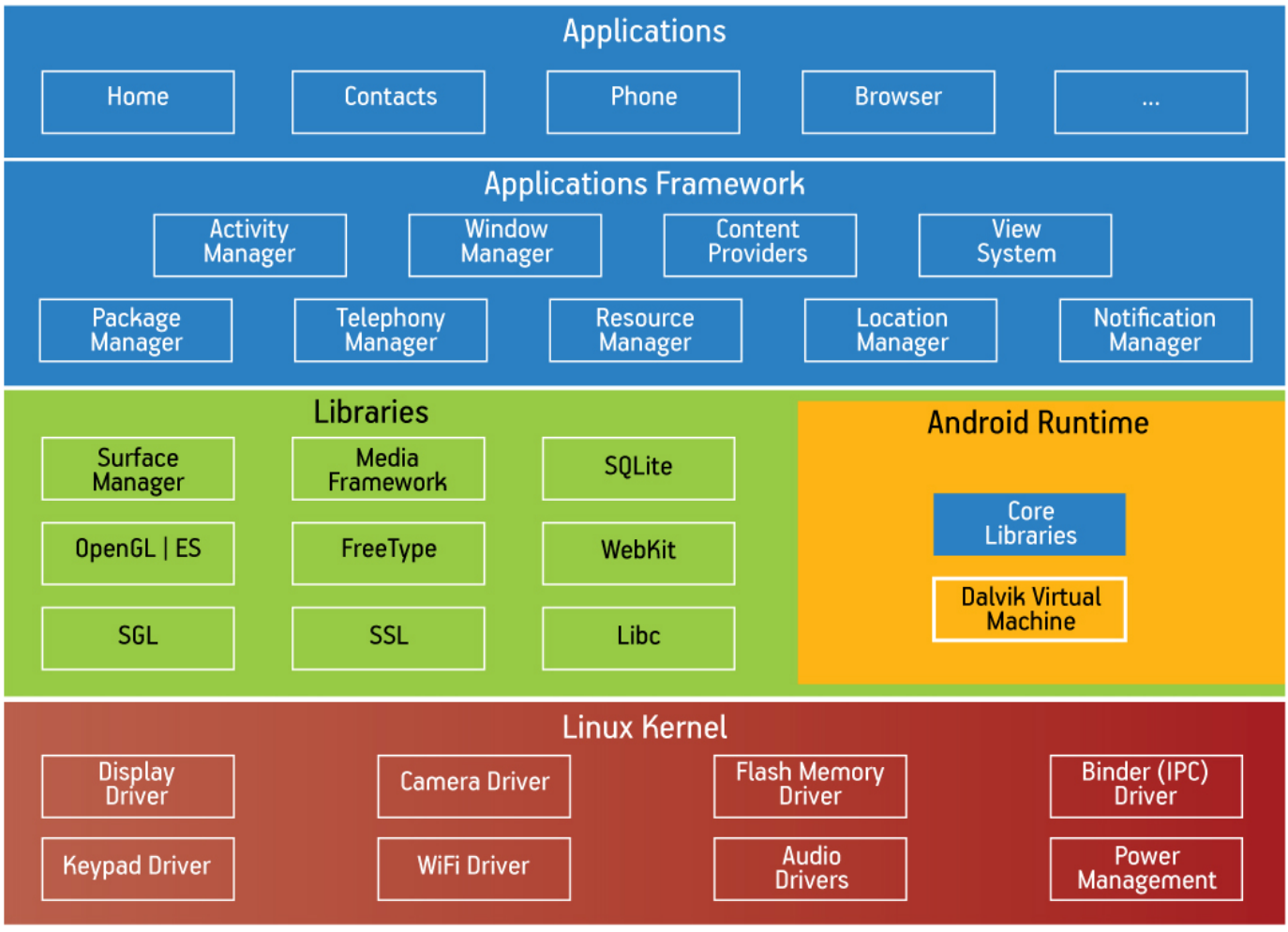

Figure 3.1 Android architecture [2]

Figure above shows the various components of the Android operating system. Each layer has different functionalities to the other layers.

The components in the Android architecture are explained below:

- Applications - The Android application is located in this layer and the user uses it. The application is developed and changed in this layer.

- Application framework - This is the main layer where the application directly interacts with and can manage the basic functionalities and can communicate with different classes, other applications or services

- Libraries - It is the layer that enables the device to handle various types of data. These libraries are written in C/C++ hardware and are specific for particular hardware.

- Android Runtime - Android runtime contains libraries native to Android and virtual machine to run the native code.

- Linux Kernel - This provides the basic system functionality. Communicates and interacts with the internal hardware and contains all the essential hardware drivers.

Chapter 4 - SYSTEM DESIGN

The system design for this application is explained using the UML diagrams. The following diagrams are used to brief the design and the flow of the activities ind detail.

- System Flow Diagram
- Use Case Diagram
- Class Diagram
- Sequence Diagram

4.1 System Flow Diagram

The given diagram is the system flow for the application and it contains the following steps,
Step 1: Configuring the user settings
Step 2: Providing all the required information that is needed as a Doctor/Nurse/Caretaker
Step 3: Updating the fall information to server and storing it in the database
Step 4: Perform the necessary action (such as call 911), if necessary

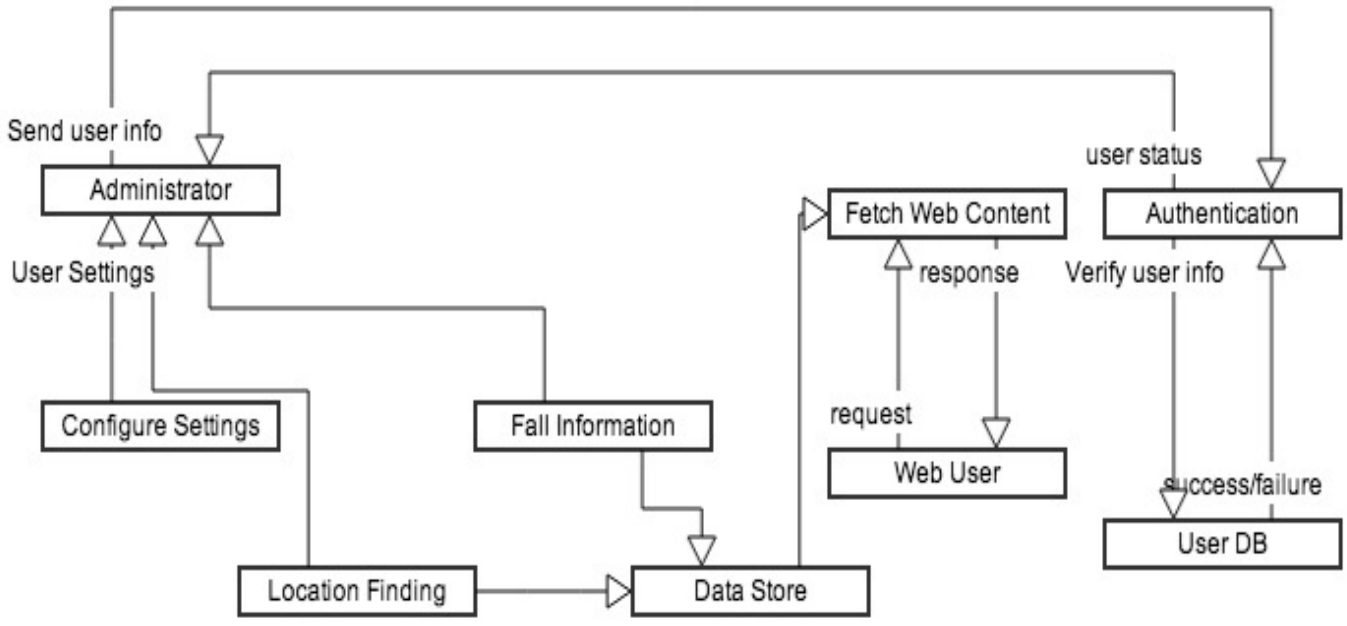

Figure 4.1 System Flow Diagram

4.2 Use Case Diagram

The important modules of the use-case diagram are categorized as the following,

1. Fall Detection
2. Location Tracking
3. Communication
4. Helpline video
5. Route Map Integration

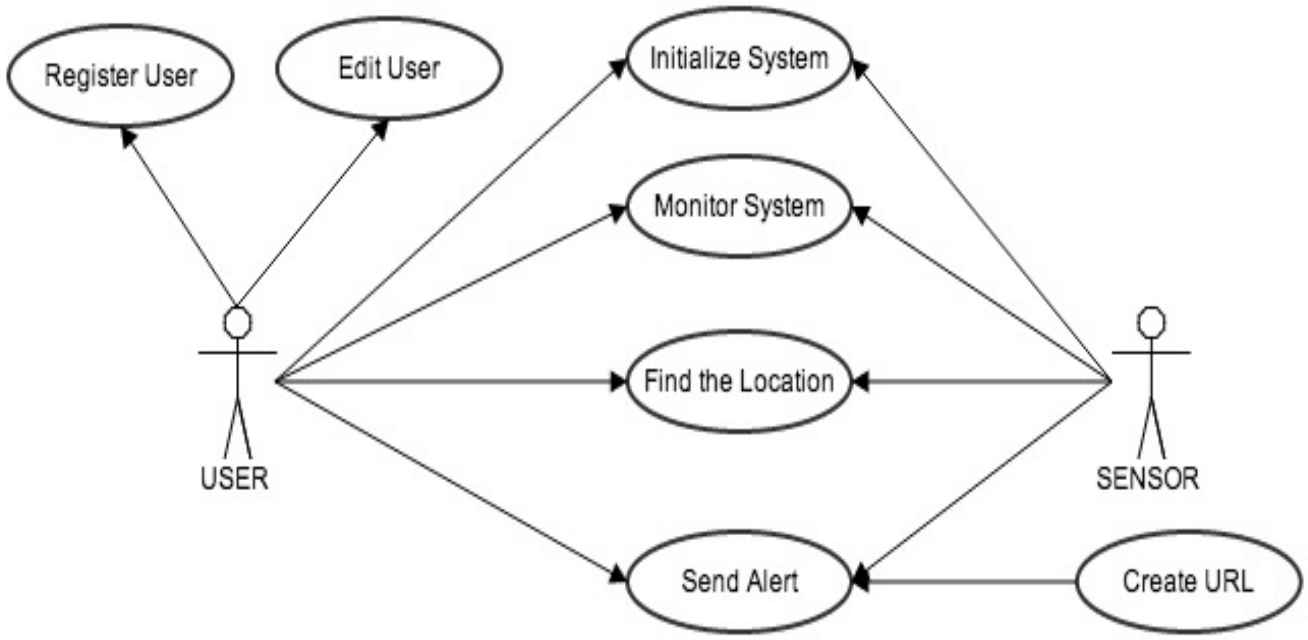

Figure 4.2 Use Case Diagram

4.2.1 Fall Detection:

The fall detection is used to monitor the elderly for possible falls, so that they can be safe at all times. It is used to detect and alert the caretakers about the area in which the fall occurred. All the Android devices are in-built with sensors like accelerometer and gyroscope; these can also be used as angle monitors. The sensors are used to predict the fall detection based on change in the angle of the mobile. The algorithm used to detect the falls is explained in Chapter 5.

4.2.2 Location Tracking:

Real-time location systems (RTLS) are used to automatically identify and track the location of objects or people in real time, usually within a building or other contained area.

4.2.3 Communication:

SMS Communication is maintained between the caretaker and the elderly that contains the fall location. The received message contains emergency alert URL based location link. Thus, when a caretaker gets an alert message, he can find the location with the help of Google maps.

4.2.4 Helpline Information:

Helpline Information contains the elderly persons information like address, caretakers numbers, blood group and the personal medical details. It will play automatically when there is a fall. This information is very useful when someone around is trying to help in that situation.

4.2.5 Route Map Integration:

If the caretaker clicks on the URL link from the message, the exact location of the fall is displayed. It then integrates this information into Google Maps through Google Maps API, which displays the position on a map. Since the positional information is retrieved every second and the maps are updated at the same frequency and a real time GPS tracking effect is achieved.

4.3 Class Diagram

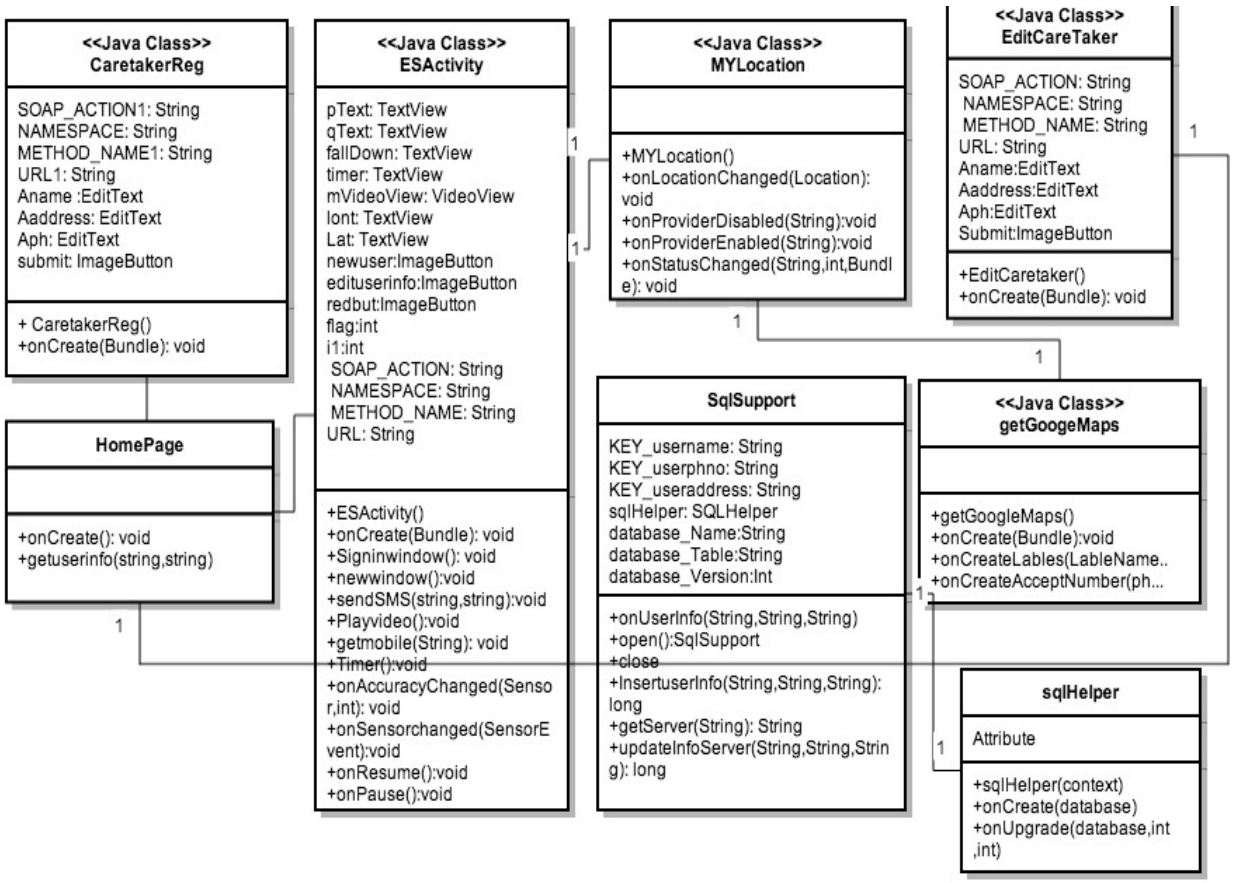

Figure 4.3 Class Diagram

A UML class diagram is used to describe the system structure and it uses classes, attributes and the relationships among classes. The class diagram is the main building block of object-oriented model. Each activity in the Android application is depicted as a class and the interaction is explained as the relationship between the classes. The figure depicts the design in the form of a class diagram.

4.4 Sequence Diagram

Sequence Diagram is used to depict the interactions of objects in sequence of time. The communication between those objects during the execution of a particular function is also depicted in the sequence diagrams. The sequence diagram that is shown below is the flow of actions performed to explain the sequence of operations in the Android application.

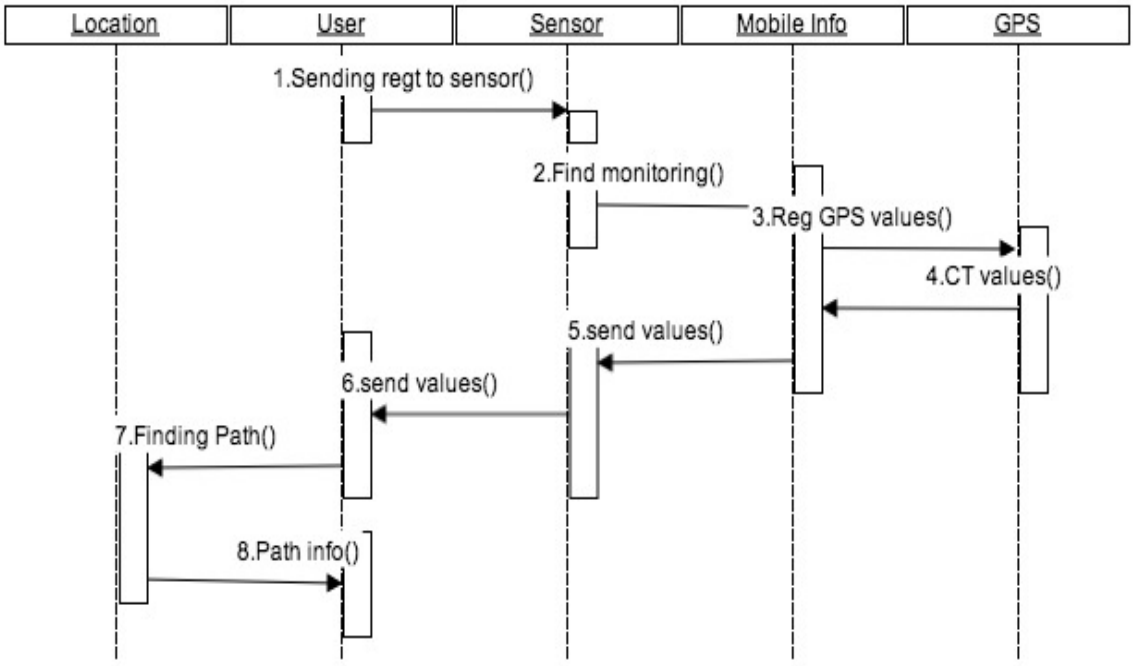

Figure 4.4 Sequence Diagram

Chapter 5 - FALL DETECTION ALGORITHM

The fall detection algorithm is discussed in detail in this chapter. This chapter also discusses the sensor inputs utilized.

For detecting a fall event, the application monitors the Accelerometer and Magnetic field sensors in background. A fall event is characterized by detection of changes similar to fall patterns in both acceleration and orientation occurring within a short interval.

5.1 Sensors

Android API device sensors are exposed to a class called SensorManager. As we need to check the acceleration and orientation changes, the SensorManager is used to verify the accelerometer and magnetic field sensors. Upon change in sensor values, an object of class SensorEvent is passed to the sensor event listener method. The SensorEvent class holds the information like sensor type, sensor's data, event timestamp, and accuracy. Following diagram depicts the coordinate system used by the SensorEvent API.

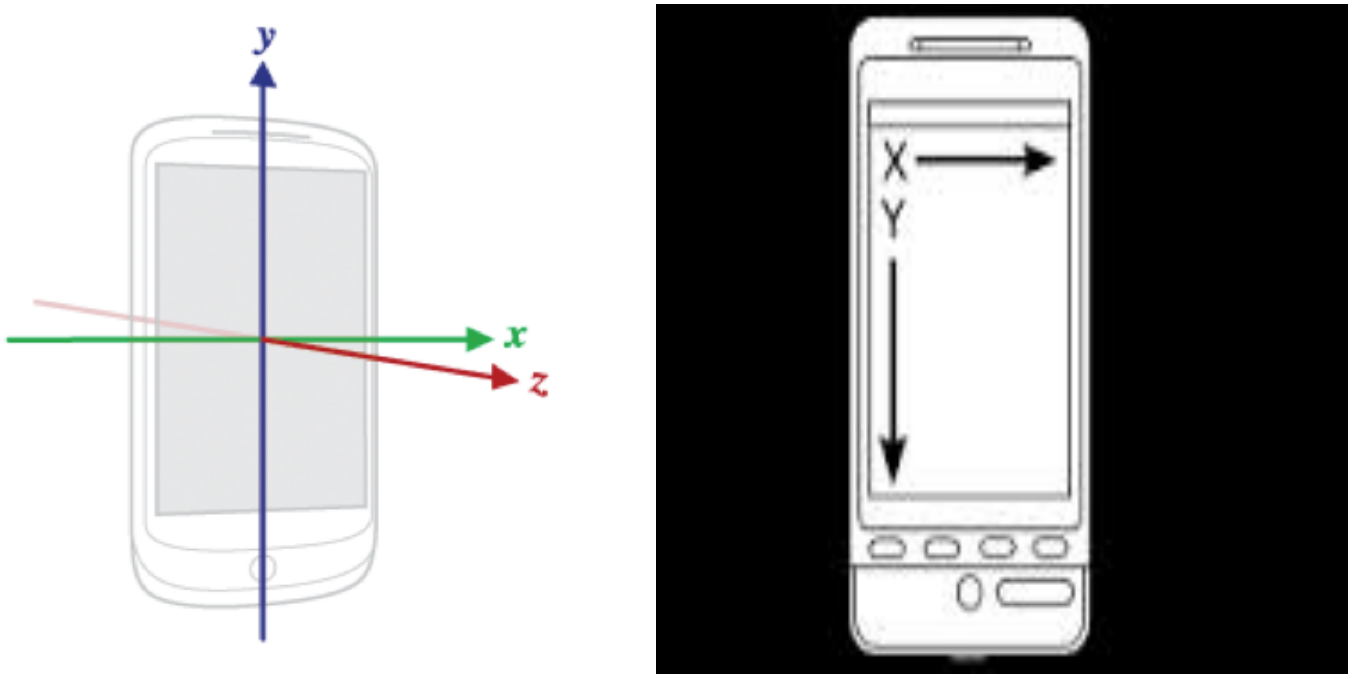

Figure 5.1 - SensorEvent API coordinate system[11][12]

As shown in the Figure 5.1 above, X axis is horizontal and points to the right, the Y axis is vertical and points up and the Z axis points towards the outside of the front face of the screen. An application subscribing to sensor events is also supposed to unregister the listeners when it no more requires the sensor monitoring. This is required to prevent draining of the battery. So, in the application, listening to sensor events is stopped when the user stops monitoring.

5.2 Acceleration:

The SensorEvent object received in the sensor listener contains an array of sensor values. These sensor values depend on the type of the sensor and for accelerometer, the values that are returned are in m/s^2

- Acceleration minus Gx x-axis
- Acceleration minus Gy y-axis
- Acceleration minus Gz z-axis

The acceleration applied to the divide is measured using the Accelerometer sensor and the measured acceleration is influenced by the force of gravity.

5.3 Orientation:

Orientation of the device is calculated using the values obtained by both accelerometer and magnetic field sensor. The sensor event listener calculates both these values and using these two sets of the values and the methods provided by the Android API, the calculation of a rotation matrix transforming a vector in device coordinate system into world coordinate system. We then calculate the device orientation from this rotation matrix

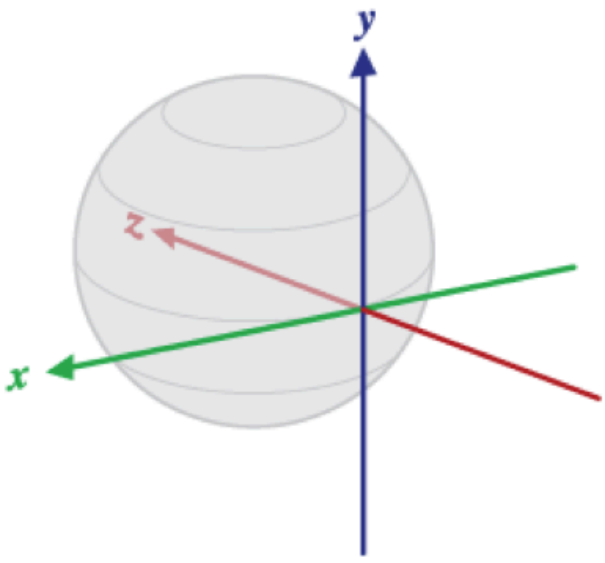

Figure 5.2 - Reference coordinate system for device orientation[13]

In the reference coordinate system

• X is defined as the vector product Y.Z (It is tangential to the ground at the device's current location and roughly points in the West direction)

• Y is tangential to the ground at the device's current location and points towards the magnetic North Pole

• Z points towards the center of the Earth and is perpendicular to the ground

All three angles above are in radians and positive in the counter-clockwise direction. During fall, user position changes from standing to supine, in a sudden manner. This change produces a variation in pitch. In our application, the pitch values form the dataset that is used for real-time classification.

5.4 Threshold-based Fall Detection

The approach is used to detect the occurrence of fall by monitoring whether the values output by sensors cross certain thresholds.

During the initial experiments involving falls, I observed that a fall produces a downward spike and then an upward one in the acceleration pattern. During a fall, the acceleration typically ranges below 6 m/s^2, and then increases to 12 m/s^2, before stabilizing around 10 m/s^2. High-impact falls produce more variations; I also observed acceleration going below 2 m/s^2during some falls. Orientation exhibits high-magnitude change, which results in the change in position from standing to supine. I decided the thresholds for fall detection based on these observations. A fall was characterized as 'acceleration falling below 6 m/s^2 and then going above m/s^2, and pitch changing by more than 75 degrees.' For fall detection, both acceleration and pitch value streams are checked.

5.5 Dynamic Programming based Approach

Dynamic Time Warping is a classical algorithm based on dynamic programming to match two time series with temporal dynamics, given the function for calculating the distance between two time samples.

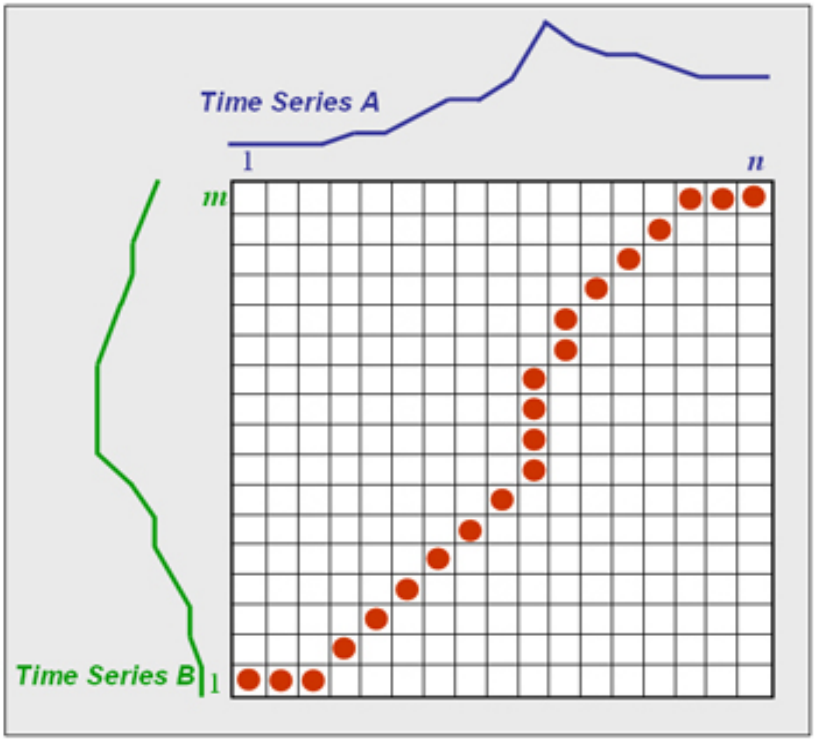

Figure 5.3 Dynamic Time Warping [14]

Let S [1...M] and T [1...N] denote the two time series. As shown in the figure 5.3, any matching between S and T with time warping can be represented as a monotonic path from the starting point (1, 1) to the endpoint (M, N) on the M by N grid. A point along the path, say (i, j), indicates that S [i] is matched with T [j]. The matching cost at this point is calculated as the distance between S [i] and T [j]. The path must be monotonic because the matching can only move forward. The minimum accumulative distance of all possible paths, or matching cost evaluates the similarity between S and T. Dynamic Time Warping uses dynamic programming to calculate the cost and to find the corresponding optimal path.

Using dynamic programming for fall detection in this application involves finding similarity between a real-time sensor data pattern and existing fall templates using Dynamic Time Warping. Matching cost <threshold identifies a fall. Calculating the matching costs of various fall patterns chose the threshold value. The Euclidean distance was used for distance calculation.

These approaches produced very good results as a fall pattern involves values corresponding to user being stationary after fall.

Chapter 6 - ANDROID FRAMEWORK COMPONENTS

This section discusses the various components of an Android application such as activity, fragments, intent and SQLite database.

6.1 AndroidManifest.xml

Every application must have an AndroidManifest.xml file in its root directory. It contains important details about the application that is essential for the system to run the code. The manifest file contains information about the activities and services in the application with intent and intent-filters. The manifest.xml also has the minimum SDK version of this application for stating its compatibility with other versions. The permissions that are required to access other API's such as Bluetooth, maps, speech, network etc. are also located in the manifest.xml.

The AndroidManifest.xml file used for the Elderly Support application is given below.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.es"
    android:versionCode="1"
    android:versionName="1.0">
  <application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".ESActivity"
          android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>

  <activity android:name="EditCaretaker"></activity>
<activity android:name="CaretakerReg"></activity>
</application>
  <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
<uses-permission android:name="android.permission.READ_PHONE_STATE"></uses-
permission>
</manifest>
```

6.2 Activity

The interaction of the user application is an important aspect, which is checked using the Activity, which contains backbone logic for front end Graphical User Interface. Each activity is associated with a XML view, which contains user interface components such as text boxes, buttons, and labels etc. that are directly viewable by user.

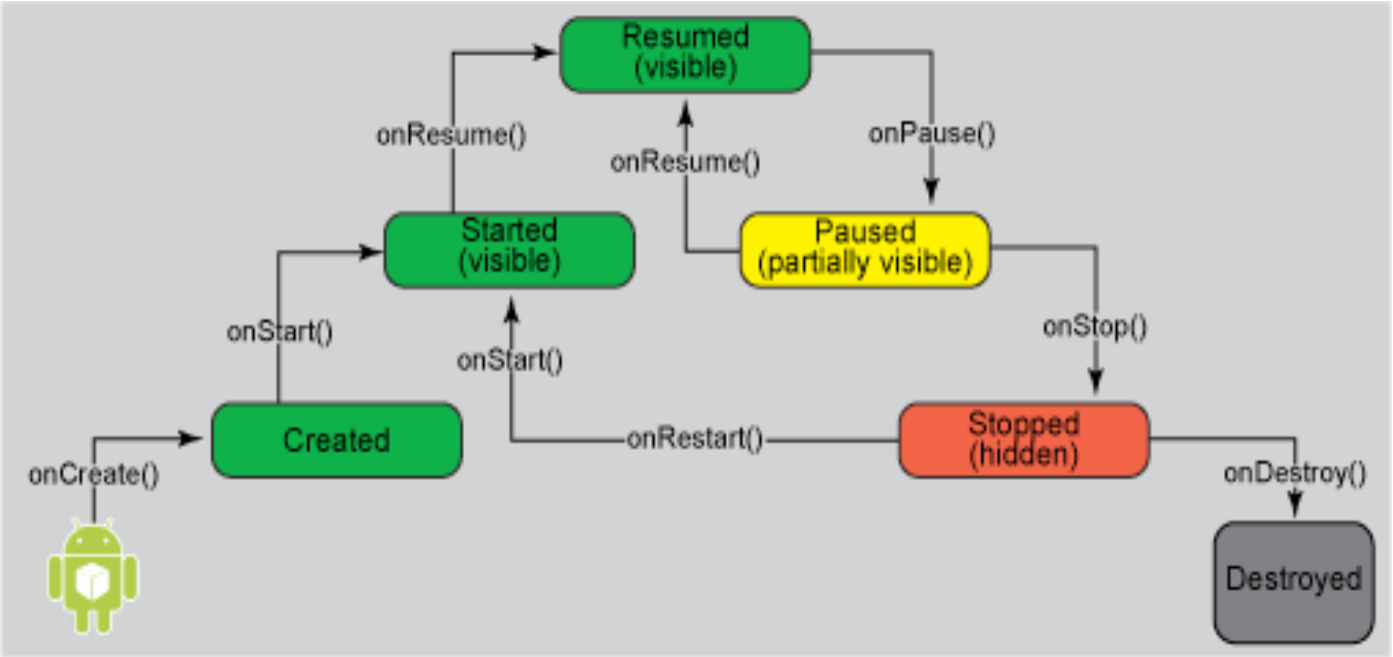

Figure 6.1 Android Life Cycle [3]

During its life cycle, an Android activity undergoes different call back methods like Create, Start, Resume, Pause, Resume, Stop and Destroy. Activity alters its display on the state change. Figure 5.1 below shows the life cycle of an activity. The sample code for activity initialization is given below,

```
 @Override
protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_create_user);
 setupActionBar();
 TextView userName = (TextView) findViewById(R.id.newuser);
 . . .
 }
```

6.3 Fragments

Fragments, as the name suggests is an Android component, which represents the partial behavior of user interface in an Activity. They are used when a need of similar portion of code in multiple activities, so fragments can be reusable. They can also be combined in single activity to build a multiple user interface and they can exist without an associated UI. They can be added or removed while the activity is running. However, even if fragment has a separate life cycle, it cannot exist independently and so it should always be in an activity.

6.4 Intents

Intent is started using startActivity() function. It has different components like action, data, category, extras and flags which can be set before starting the intent using startActivity(). There are two major types of intents - implicit intents and explicit intents. Implicit intents are those, which are communicating with other external applications instead of components specified in the application. Explicit intents are those, which calls the components in the application by its activity name. Explicit intents uses the activity names registered in the AndroidManifest.xml.

6.5 SQLite Database

Android applications, which require relational database to store data can use different databases. Android platform has a built-in SQLite database engine, which is self-contained, server less and requires no configuration. The database is created continuing the SQLiteOpenHelper class. The database after creation is stored in device memory, which makes the database accessible to that application. The onCreate() function of the SQLiteOpenHelper class is overridden to create the database.

Chapter 7 - IMPLEMENTATION

The Android application is developed by creating the project on Eclipse IDE with the Android SDK plug-in installed in it. The Emulator and Logcat are tools that aid in development of Android application to test and debug the application.

The main objective of Elderly Support application is to provide an easy to use interface for the caretakers to be notified when a fall occurs.

Additionally, the other main features of the app include:

- Obtaining the current location
- Showing the route map
- Send details to phone via SMS
- Send details to Email

The Android application is developed using the Eclipse Juno IDE. Android SDK Version 4.4 has been installed for this application.

The business logic is written in Java. The user interface is developed using XML. The user interface is designed to be user friendly to all the users and it doesn't require manual guidance.

The total lines of code in this application are 3729 lines including the Java and the xml files. The split up of the lines of code is given as follows

LANGUAGE	LOC
Java	2746
XML+ Manifest	983

Table 7.1 Lines of Code (LOC)

Debugging of the application throughout the development is done using Dalvik Debug Monitor Server (DDMS). DDMS provides port-forwarding services, screen capture on the device, thread and heap information on the device, logcat, process. DDMS is also used to verify the location-based services that are used in the application.

7.1 Graphical User Interface

The user interface is designed to be more simple, accessible, understandable and consistent with all kinds of users. The GUI has been designed to be compatible with both landscape and portrait mode. This application is designed targeting Android mobile users rather than tablet users as this type of applications are most widely used in motion and the mobile devices are more comfortable for doing so. The navigation between all three main functionalities of this application makes it easy to use and requires no navigational instructions.

7.1.1 Login Screen

Login screen is displayed to user when the application is launched. The Android-manifest file indicates this activity as the starting point of Elderly Support application.

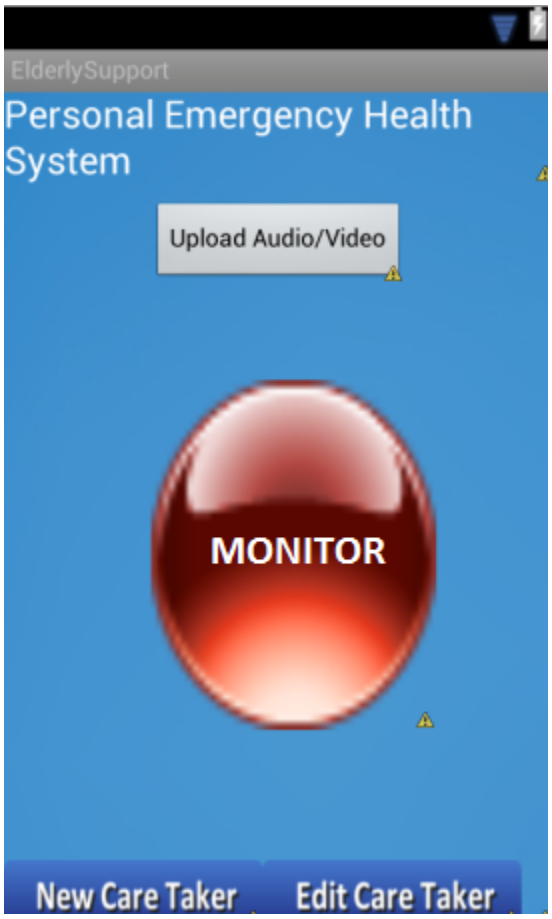

Figure 7.1 Login screen

7.1.2 Add a new user Screen

First time users need to create an account by providing basic information like name, address, phone number and valid email address. Basic validation rules are applied while accepting user details in order to avoid spam information coming into the application.

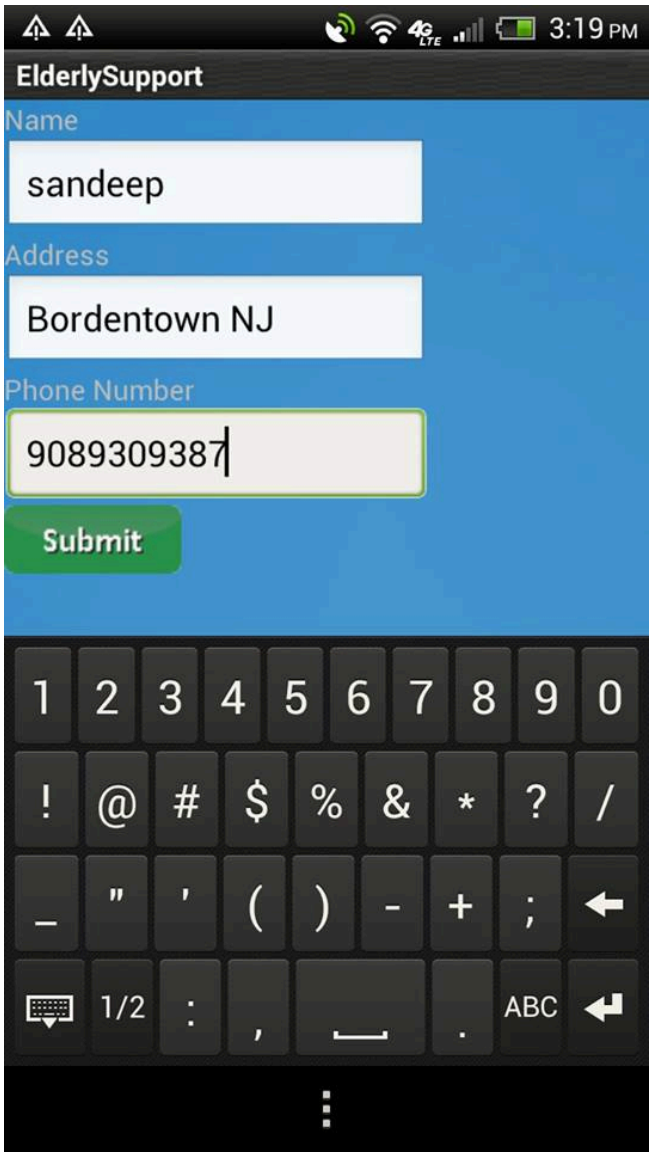

Figure 7.2 Add new user screen

7.1.3 Edit user information screen

Users information that needed to be edited or changed is done in this screen. Once the change has been made, a notification message is given.

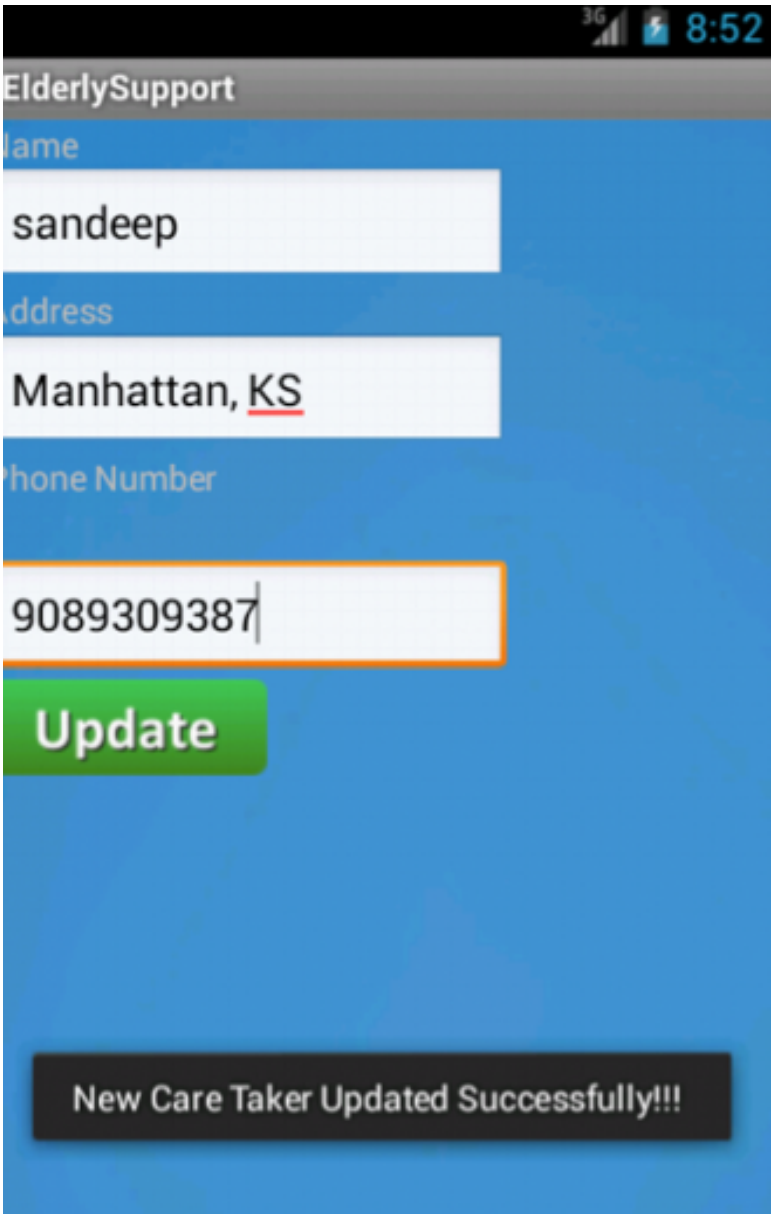

Figure 7.3 Edit user information screen

7.1.4 Alert Generation Screen

Figure 7.4 Alert generation - Timer

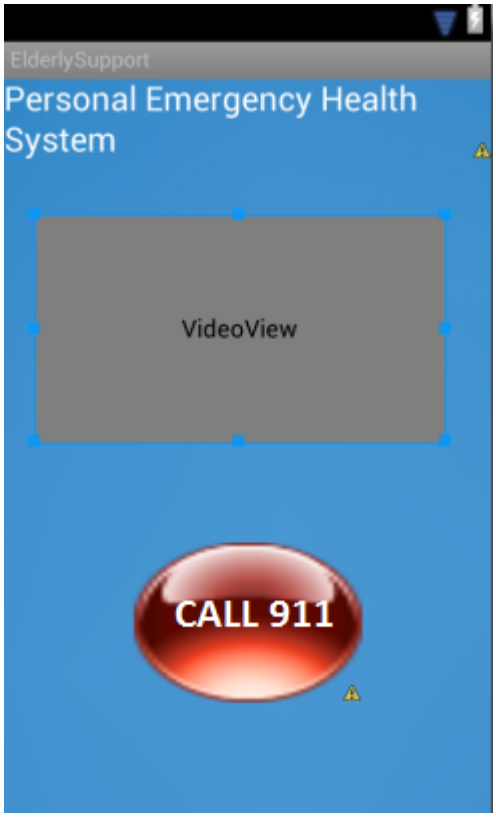

Figure 7.5 Alert Generation - Video

7.1.5 Received Text Message

If the timer reaches zero and the person doesn't get up, it sends an automatic alert to the caretaker and the message contains the location in which the fall occurred. The figure given below shows the message and the exact location at which the fall occurred.

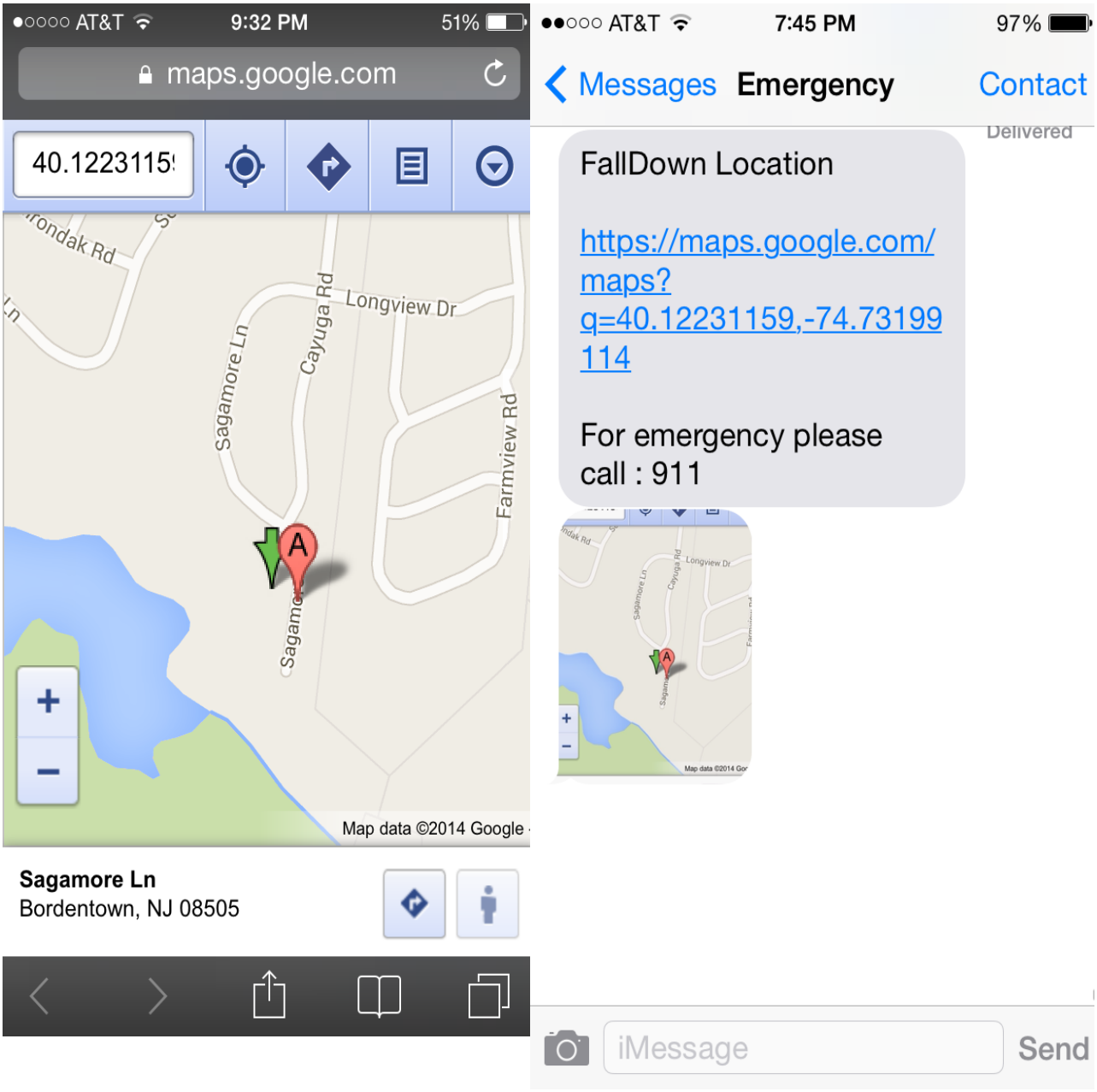

Figure 7.6 Received text message

Chapter 8 - TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design, and code generation. Once the source code has been generated, software must be tested to uncover as many errors as possible. A test case is one that has high probability of finding a yet undiscovered error. A successful test is one that uncovers as yet undiscovered error.
Android framework provides a default logging and debugging tool – DDMS (Dalvik Debug Monitor Server). The DDMS is default with the Android SDK plug-in when the Android application development is installed on Eclipse IDE. The errors that are encountered are recorded as log messages in the Logcat. The aspects in which the Android application is tested are discussed in this chapter.

8.1 Unit Testing

Instead of testing the system as a whole, unit testing focuses on the modules that make up the system. Each module is taken up individually and testing for the correctness in the coding and logic. Errors resulting from interaction of modules are initially avoided. The following tests are run manually on HTC one X and Samsung Galaxy tab 2 for its correctness.

Sr.No	Test Case	Expected Result	Result
1	On load of startup screen	Display the screen with all initial information needed	Pass
2	On click of new user	Diaplay a form to get new user information	Pass
3	On click of edit user	Display information needed to be changed for a particular user	Pass
4	Fall occurred	Start the timer from 30 seconds	Pass
5	Alert message	When the timer reaches 0, an alert message is sent to the caretaker	Pass

Sr.No	Test Case	Expected Result	Result
6	on touch if The red button after the fall	The timer is reset	Pass
7	Alert message sent	After the alert message is sent the video with patient information plays	Pass
8	On click of submit to add new user	If the information entered is not valid, give a floating message to re-enter	Pass
9	On click of edit to edit a previous user information	If the user is not in the database, error message is displayed	Pass
10	On click of edit to edit a previous user information	Changed the information of the particular user	Pass

Table 8.1 Unit Test Cases

8.2 Integration Testing

It tests for the errors resulting from integration modules. One specific target of integration testing is the interface: whether parameters match on both sides as to type, permissible ranges and meaning. In this all modules tested separately would be put together and tested for producing the ultimate result of the system. The main emphasis during this testing will be on the interface between the modules.

Class	Test Case	Expected Result	Observed Result	Result
Navigation	Action call	Corresponding action is called	action is invoked	Pass
Communication	Between Activities	Information sent and resulting action	information retrieved	Pass
Communication	Between the server and the device	Stores the details in the database	Data stored in the DB	Pass

Class	Test Case	Expected Result	Observed Result	Result
Communication	Between the server and the device	get the details from the database and use	information retrieved and passed to corresponding action	Pass

Table 8.2 Integration Test Cases

8.3 Compatibility Testing

The elderly support android application is developed such that it can be run on a range of Android devices with different display. This application is tested in HTC one X and Samsung Galaxy Tab 10.1. The application is tested in portrait and landscape modes and the figures given below are the difference in both smart phones and tablets.

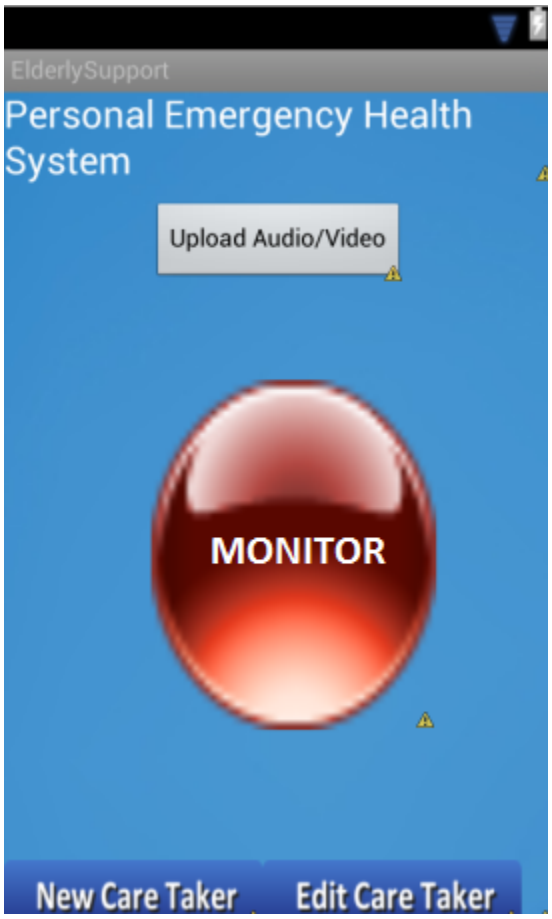

Figure 8.1 Portrait View of homepage

Figure 8.2 Landscape View of Emergency

8.4 Battery Consumption

Following table presents a measure of battery consumption for the applications developed. The application was tested with the time for the battery level to reduce from 100% to 80%, with/without the application running. Phone's usage conditions for these measurements included audio playback, voice calls, and moderate Internet access over both Wi-Fi and 3G networks.

Application	Time taken for Battery to reduce (20%)
Phone's normal use, no application running	109 minutes
Data collection applications, with sensor data upload and phone's normal usage	96 minutes
Elderly Support application running - Fall detection, Alert and the phone's normal usage	89 minutes

Table 8.3 Battery Consumption

Chapter 9 - CONCLUSION

This is my first attempt in developing an Android mobile application. This gave me the basic understanding of development and challenges involved in mobile application development. The main aim of the project is to provide an easy, user-friendly application to help the elderly during a fall. The application has been implemented and tested on real devices. Android being an operating system that is customizable and open source, a new application can be developed according to one's requirements. Also, the documentation available online for Android application development makes it very helpful to design an Android application.

This project enabled me to learn Android application development and It helped me in understanding the working of GPS, in-built sensors and Google Maps V2 API for showing locations on Google Maps. In this project, I have also learnt to measure the performance of an Android application.

Chapter 10 - FUTURE WORK

The application can be improved in many ways and can be extended to support devices such as tablets and iOS devices. Following are some of the possible extensions to extend the system's capabilities by incorporating new services. These services include

- Embedding a belt measuring heart rate as an external sensor
- Integrate a gyroscope sensor instead of an orientation sensor, for more accurate results
- Integration of social networks to alert senders
- Integrate public agency to alert senders
- Add a system administrator feature.

Chapter 11 - REFERENCES

[1] Android basics. http://developer.android.com/training/index.html

[2] Android Architecture [Online]
[Author: Ville-Veikko Helppi] [Posted : April 15, 2010]
http://www.techdesignforums.com/practice/technique/android-beyond-the-handset/

[3] Android Life Cycle [Online]
[Author: Andrew Glover] [Posted : May 28th, 2013]
http://www.ibm.com/developerworks/mobile/library/j-mobileforthemasses3/index.html

[4] Android Architecture Description
[Posted : January 2nd, 2013]
http://www.edureka.in/blog/beginners-guide-android-architecture/

[5] Android System Architecture. Smieh. 2012, Anatomy Physiology of an Android.

[6] Google Maps V2 API.
https://developers.google.com/maps/documentation/android/

[7] Android code snippets [Online]
http://stackoverflow.com/

[8] J. R. Kwapisz, G. M. Weiss and S. A. Moore, "Activity Recognition Using Cell Phone Accelerometers," in Fourth International Workshop on Knowledge Discovery from Sensor Data, 2010.

[9] L. Sun, D. Zhang, B. Li, B. Guo and S. Li, "Activity Recognition on an Accelerometer Embedded Mobile Phone with Varying Positions and Orientations," in Ubiquitous Intelligence and Computing, 2010.

[10] K. Cline, "The History of Android," [Online].
Available: http://venturefizz.com/blog/ history-android.

[11] what is Android Sensor Manager ? [Online]
[Posted : February 11th, 2013]
http://www.electronicsweekly.com/eyes-on-android/what-is/what-is-the-android-sensor-manager-2013-02/

[12] Understanding Android Screen Densities [Online]
[Posted : March 20th, 2013]
http://tekeye.biz/2013/android-dpi-dip-dp-ppi-sp-and-screens

[13] Android : Gyroscope Basics [Online]
[Author : Kaleb Kircher]
http://www.kircherelectronics.com/blog/index.php/11-android/sensors/15-android-gyroscope-basics

[14] Dynamic Time Warping Algortihm [Online]
http://cst.tu-plovdiv.bg/bi/DTWimpute/DTWalgorithm.html